Machine Learning Journal, 20, 1–17 (1995)

Evaluation and Selection of Biases in Machine Learning

DIANA F. GORDON gordon@aic.nrl.navy.mil
Naval Research Laboratory

MARIE DESJARDINS marie@erg.sri.com
SRI International

Editor: Thomas G. Dietterich

Abstract. In this introduction, we define the term *bias* as it is used in machine learning systems. We motivate the importance of automated methods for evaluating and selecting biases using a framework of bias selection as search in bias and meta-bias spaces. Recent research in the field of machine learning bias is summarized.

Keywords: bias, concept learning

1. Introduction

This special issue of *Machine Learning* focuses on the evaluation and selection of biases. The papers in this issue describe methods by which intelligent systems automatically evaluate and select their own biases, and tools for analyzing and testing various approaches to bias selection. In this paper, we motivate the importance of this topic. Since most readers will be familiar with supervised concept learning, we phrase our discussion within that framework. However, bias as we present it here is a part of every type of learning.

We outline a framework for treating bias selection as a process of designing appropriate search methods over the bias and meta-bias spaces. This framework has two essential features: it divides bias into representational and procedural components, and it characterizes learning as search within multiple tiers. The sources of bias within a system can thus be identified and analyzed with respect to their influence on this multi-tiered search process, and bias shift becomes search at the bias level. The framework provides an analytic tool with which to compare different systems (including those not developed within the framework), as well as an abstract formalism and architecture to guide the development of new systems.

We begin by defining what we mean by the term *bias.* Next, we explain why the selection and evaluation of biases is a critical task for intelligent systems. We then describe our search-based framework for bias selection. Finally, we survey recent research in this field, using the concepts developed in our framework to guide the discussion.

2. What is a bias?

Mitchell [23] defines *bias* as "any basis for choosing one generalization over another, other than strict consistency with the instances." We broaden this definition to include any factor (including consistency with the instances) that influences the definition or selection of inductive hypotheses.[1] There are two major types of bias: representational and procedural. Background (e.g., task) knowledge has sometimes been considered to be a bias as well [17]. However, since knowledge has the supportive role of providing information to select a representational or procedural bias, here we do not consider it to be a bias *per se*.

A *representational bias* defines the states in a search space. Typically, this search space is the space of hypotheses. A representational bias specifies a language (such as first-order predicate calculus or a restriction to disjunctive normal form (DNF) expressions), an implementation for this language (e.g., DNF can be implemented using rules or decision trees), and a set of primitive terms (the allowable features, their types, and their range of values), thereby defining the set of states.

Representational bias can be characterized along several axes, including *strength* and *correctness*. According to Utgoff [36], a strong representational bias for the hypothesis space implies a small hypothesis space; a weak representational bias implies a large hypothesis space. A representational bias is considered correct if it defines a hypothesis space that includes the target concept; otherwise, it is incorrect.

The computational learning community has also explored the issue of bias strength from a formal, analytical perspective. Vapnik and Chervonenkis [38] define a measure of the size of a bias defined by a given representation, called the VC-dimension. Blumer et al. [7] use the VC-dimension to provide bounds on the number of examples required for any consistent learning algorithm to approximate the target concept with high confidence.

A *procedural bias* (also called *algorithmic bias* [26]) determines the order of traversal of the states in the space defined by a representational bias. Examples of procedural biases include the beam width in a beam search and a preference for simple or specific hypotheses. Occam's Razor [6] and the Minimum Description Length Principle [9] [33] provide formal motivations for why a preference for simple hypotheses works well theoretically. However, they leave the question of a practical implementation open, and appropriate representational biases and search heuristics that find simple hypotheses are still required.

Note that biases may interact; a procedural and a representational bias might interact synergistically or conflict. Few researchers have studied bias interactions (see Cardie [8] and Cobb [11] for exceptions). Hopefully, future work will explore this important topic.

Both representational and procedural biases can be evaluated (empirically or analytically) by determining the effect they have or are expected to have on learning performance. Bias selection involves using the results of this evaluation process to choose a bias (or a sequence of biases) for use during learning. *Shifting* bias refers to the special case where bias selection occurs again after learning has already begun.

In this case, the system may simply choose the next bias in a bias sequence that was established prior to learning, or it may use the results of the learning performed so far to evaluate potential alternative biases. In either case, there will be a need to incorporate the knowledge that has already been learned to initialize and guide the search through the space defined by the new bias.

3. Bias selection

Here is a very simple example that motivates the importance of selecting an appropriate representational bias, in this case by selecting a subset of the available features for expressing hypotheses. Suppose that the target concept is a description of blocks that are stable when placed on a table, and further suppose that the shape of an object determines its stability (though the learner is not told this). The perceptible features of the blocks are "size," "color," and "shape." Two examples are given to the learner: a small, blue cube that is a positive instance and a small, red sphere that is negative (it rolls off the table). If the bias selected is that "size" and "color" are preferred to "shape" in forming concept descriptions, the system might form the hypothesis:

h_1. Small, blue blocks will be positive and small, red ones negative.

If the bias selected is that only "shape" should be used, the following hypothesis might be formed:

h_2. Cubes will be positive and spheres negative.

Both of these hypotheses are consistent with the training examples seen so far. Suppose that a small, blue sphere is now observed. Then h_1 will predict that it will be stable, and h_2 will predict that it will not be stable. In this case, h_2 will be the better predictor of the concept of stability, demonstrating that judicious bias selection can improve the predictive accuracy of the learner. Judicious bias selection can also improve the learner's ability to achieve other performance goals in addition to accuracy, such as efficiency (i.e., a reduction in time and/or space complexity) or readability.

In more complex, real-world domains, with potentially hundreds of features and many sources of data, bias selection becomes even more critical: not only can choosing the wrong feature set make learning a correct concept impossible or computationally overwhelming, but other aspects of bias become important as well. For example, using inappropriate heuristics to search the hypothesis space may cause the learning system to get stuck at local maxima or to take so long to find the correct hypothesis that the time for the performance task has expired before the concept is learned.

Various methods can be used to select biases that improve learning. The framework presented in Section 5 considers bias selection as searching a space of learning biases. In this framework, as in any search paradigm, operators for generating states in the space (biases) and evaluation techniques to determine which state to explore (bias to select) are required. Evaluation of biases may use various forms of knowledge about the learning context, including background knowledge about relevance, information about costs of learning and making predictions, and knowledge about the performance of the learning methods being used.

4. Bias evaluation

Bias evaluation methods are needed to form a basis for bias selection. The cost and quality of bias evaluation methods will have a large impact on the overall benefits reaped from bias choices. Typical categories of evaluation methods include generate-and-test (online and empirical) and theoretical studies (offline and analytical). Generate-and-test methods are valuable for gathering knowledge in knowledge-poor situations. Predictive theoretical analyses are also valuable when they do not make too many simplifying assumptions. The knowledge gained from generate-and-test and analytical evaluations may be kept offline. Chrisman [10] describes a method for analyzing learning performance to identify inappropriate biases (this can be thought of as the "test" stage of a generate-and-test evaluation method).

Just as search heuristics are used at the hypothesis space level to find a good hypothesis, the use of heuristics is a potentially efficient method for bridging the gap between the evaluation and selection of biases. Heuristics compile the knowledge gained from previous bias evaluations to be used for future bias selection. Examples of heuristics (which are also biases - see below) are "Prefer simple hypotheses" or "Order the biases from strongest to weakest." If these heuristics have preconditions (e.g., "If the data is noisy and the goal is to improve predictive accuracy, then prefer simple hypotheses"), then we can associate regions of expertise with biases. In other words, we can identify the "best" bias for each given set of problem space characteristics (e.g., the quality of the data, the type of attributes, or the contents of the current hypothesis) and performance goals. VBMS [28] is an example of this approach. In VBMS, the domain-independent heuristics are learned from "training examples" that are themselves inductive learning problems. Brodley's MCS (this issue) is a more recent heuristic-based system in which the heuristics are manually generated.

Many of the papers in this special issue could be considered as providing steps toward the carving out of regions of expertise for biases. Recently, there has been a great deal of discussion regarding "no-free-lunch" theorems about induction [31], [40]. These results state that when performance is averaged uniformly over all possible problems, one learner cannot be better than another. Nevertheless, these results still allow the possibility of one learner being better than another for a particular distribution of problems. Therefore, the identification of regions of expertise of biases remains a critical task to address.

5. A search-based framework for bias selection

A *static* bias is established when learning begins and remains fixed. A *dynamic* bias can be altered during learning, requiring bias shift. Early machine learning systems used only static bias. Around 1986, a keen interest in dynamic bias developed; many systems have since been implemented that utilize methods for shifting bias. The primary reason for shifting bias is that if the knowledge for bias selection is not available prior to learning, this knowledge can be gathered during learning, thereby enabling the system to improve its learning online with a better bias once such a bias has been found. In this special issue, we hope to give the reader insights into the advantages and disadvantages of particular static biases and bias shifting methods.

Michalski [22] and Rendell [26] both show how bias shifting can be viewed as a search through a space of biases. Figure 1 presents our framework, which has three search tiers for a bias shifting system (like the framework in Rendell [26]). The lowest (first) tier in Figure 1 represents the inductive learning process, which can be viewed as a search through a space of inductive hypotheses.[2] The representational and procedural biases for the hypothesis space can be chosen either statically or dynamically. If the latter, then search is performed in the next higher (second) tier, which represents the bias space. At this level, Figure 1 shows two search spaces—representational bias space and procedural bias space. Although there are many dimensions (e.g., the choice of language, the choice of feature values) along which it is possible to make representational and procedural bias choices for defining the first-level space, for clarity we show only one procedural and one representational dimension, or search space, at the second tier.

A state in the representational bias space is a representational bias for the hypothesis space, i.e., a definition of the states in the hypothesis space. A state in the procedural bias space is an order for searching the hypotheses in the hypothesis space. At the third tier, the *meta-bias space* contains representational and procedural meta-biases for defining and ordering search in the bias space. Current bias shifting systems perform search in the first two tiers; some also perform search in the third tier. We are not aware of any systems that search in more than three tiers.

Our framework is related to both Rendell's conceptual framework [26] and the diagram in Provost [25]. However, our framework is more general than Provost's and clarifies some of the issues raised by Rendell. For example, Rendell describes the three tiers and also the representational/procedural (which he calls "algorithmic") bias distinction, but does not explain how this distinction extends easily to meta-bias space.

The question arises, why bother with multiple levels when a single level, "flat" system could produce the same behavior as the tiered system? After all, a bias shifting system is itself just another bias. The main reason for keeping multiple levels is to reduce the system engineering and knowledge engineering costs. When system engineers embed implicit biases within their hypothesis space search heuristics, they can cause subsequent problems for themselves. If these heuristics are faulty, the engineers are then faced with a highly complex debugging process that does not enable clean separation between bias and hypothesis considerations. During the design process, a multi-tiered system allows flexible system design, while maintaining conceptual simplicity. For example, one could design a k-DNF algorithm for the hypothesis level and search for k at the bias level (second tier). It would be easier to switch from k-DNF to k-CNF at the hypothesis level, for example, than if the search for k were embedded in the hypothesis space search heuristics. Cross-validation and related methods for meta-control of the hypothesis search process can also be thought of as search at a second tier of abstraction.

Furthermore, knowledge plays a key role in bias evaluation and selection. With multiple levels, knowledge engineers can enter their domain-specific background knowledge into the system more easily and more succinctly than with a flat system. For example, suppose you have a problem in which each instance has 200 features, with 30 possible values for each feature. Furthermore, suppose the knowledge engineer (a domain expert) knows the 50 relevant features out of the 200 but does not know the target concept. Feature relevance knowledge is a bias expressible in the second tier of Figure 1. If feature relevance can be represented explicitly within the system, then the engineer has few changes to make: the 50 relevant features can be tagged with their relevance. If feature relevance is not represented explicitly, the engineer must either tag all relevant hypotheses or else edit the code to generate only relevant hypotheses. Editing code can be tedious, and tagging hypotheses could entail a great deal of work, particularly if the hypotheses are expressed in the low-level language of the feature values (in which case there are 30^{50} relevant hypotheses covering single instances).

5.1. Defining the search space

We formalize this discussion as follows (see Figure 1). Let S be any search space. A search space, S, is actually a pair, $(l(S), p(l(S)))$, corresponding to the representational and procedural biases for the space. Formally, the representational bias $l(S)$ specifies the language that defines the states in a search space; however, by specifying a language we are implicitly defining the set of states expressible in that language. Therefore, for simplicity we refer to the representational bias of S as simply the set of all states in space S, where state i is denoted $state_i(S)$.

Once the representational bias, $l(S)$, has been chosen, we select a procedural bias for that particular set of states $l(S)$. The procedural bias consists of two parts, an accessibility mapping ϕ and a partial ordering θ over the states; i.e., $p(l(S)) = (\phi(l(S)), \theta(l(S)))$. The accessibility mapping consists of operators, or algorithms, that map from a state in the space to a successor state. The partial ordering, which is determined by an evaluation function, structures the hypotheses in a lattice that, when used in conjunction with a particular algorithm or set of search operators, induces an order for state space traversal. This evaluation function, when applied to states that are biases, is what we mean by "bias evaluation" in this paper. Bias evaluation may be done offline or online, and may be accompanied by heuristic preconditions. The induced order for state space traversal corresponds to our notion of "bias selection."

Suppose there are n states in $l(S)$, $state_i(S)$, where $1 \leq i \leq n$. From each state $state_i(S)$ there may be associated operators. Suppose there are m_i operators $op_k(state_i(S))$ where $1 \leq k \leq m_i$. Each of these operators maps $state_i(S)$ into a next state, i.e., $op_k(state_i(S)) = state_j(S)$ where $1 \leq j \leq n$. Then $\phi(l(S)) = \{op_k(state_i(S)) \mid 1 \leq i \leq n,\ 1 \leq k \leq m_i\}$. Thus $\phi(l(S))$ is an accessibility mapping over the states in $l(S)$. $\phi(l(S))$ may be a many-to-many mapping.

$\theta(l(S))$ is a partial ordering relation, determined by an evaluation function, over the states in $l(S)$. $\theta(l(S))$ denotes a relation $state_i(S) \preceq state_j(S)$ over pairs of states in S.

Let S^* be the (possibly infinite) set of all possible states in the universe that are candidates for being elements of S. Let L^* be the set of *all* possible candidates for the representational bias $l(S)$, i.e., L^* is the set of all subsets of S^*. To implement a tractable bias shifting system, we select a finite $L(S)$, where we define $L(S) \subseteq L^*$. $L(S)$ is the search space at the next tier up from S (see Figure 1) that defines a set of representational candidates for S, i.e., a set of candidates for $l(S)$. Let us abbreviate $state_i(L(S))$ with $l_i(S)$. Then for some integer r, $L(S) = \{l_i(S) \mid 1 \leq i \leq r\}$. Likewise, we let P^* be the set of *all* possible candidates for $p(l(S))$ and define $P(l(S)) \subseteq P^*$. $P(l(S))$ is the search space at the next tier up from S (see Figure 1) that defines a set of procedural candidates for $l(S)$, i.e., a set of candidates for $p(l(S))$. Let us abbreviate $state_j(P(l_i(S)))$ with $p_j(l_i(S))$. If we have chosen $l_i(S)$ as the representational bias $l(S)$, then for some integer q_i, $P(l(S)) = P(l_i(S) = \{p_j(l_i(S)) \mid 1 \leq j \leq q_i\}$. Once we have selected a representational and procedural bias for S, the search space S is fully defined.

Consider how these formal definitions apply to the lowest tier of Figure 1. Suppose S is a hypothesis space, H. Then each $state_i(H)$ is a single hypothesis, h_i, and $op_k(h_i)$ maps hypothesis h_i to a candidate successor hypothesis. Let H^* be the set of all possible hypotheses in the universe. Then L^* is the set of all subsets of H^*, and $L(H) \subseteq L^*$, i.e., $L(H)$ is a set of candidate sets of hypotheses. $L(H)$ is the set of representational biases being considered for $l(H)$. Each $l(H)$ can be viewed as the choice of a particular language for expressing the hypotheses. For example, $l(H)$ might be restricted to only hypotheses described by a single feature. The procedural bias $p(l(H))$ consists of the accessibility and ordering components. An example of an accessibility mapping $\phi(l(H))$ is a set of generalization and specialization operators that map hypotheses to each other. An example of a partial ordering $\theta(l(H))$ is one that expresses a preference for more general hypotheses. Note that it may be possible through the accessibility mapping to move from a state with a higher preference to one with a lower preference. Therefore, accessibility and the partial ordering need not coincide. Finally, P^* is the set of all possible mapping-ordering pairs over the hypotheses in $l(H)$, and $P(l(H)) \subseteq P^*$. We can likewise define $L(P(l(H)))$, $P(l(L(H)))$, and so on.

5.2. Searching the bias space

Let us consider some examples of search spaces in the second tier, i.e., $L(H)$ and $P(l(H))$ of Figure 1. One common method for searching the representational bias space is *constructive induction,* which uses feature constructors to move from a stronger to a weaker bias. Constructive induction is typically done to increase the likelihood that the representational bias $l(H)$ is correct, but can also improve efficiency [27]. In the example introduced previously, the features for learning about stability are "size," "color," and "shape." If the shape of an object determines its stability, but the initial hypothesis language does not include the feature "shape," then we can weaken the bias by adding this feature to the hypothesis language so that our bias becomes correct and the target concept can be learned. Alternatively, we might have operators that strengthen the bias by removing features. In Figure 2, for example, we see a search space $L(H)$. The first state of $L(H)$, $l_1(H)$, includes all hypotheses that are expressed using only the features "size," "color," and "shape." (The complete representational bias would also specify the language for constructing hypotheses from the feature set, e.g., decision trees.) A feature removal operator, $op_1(l_1(H))$, takes the system to another state, $l_2(H)$, that includes all hypotheses that are expressed using only the features "size" and "color." Note that $l(H)$, the current representational bias for H, was originally equal to $l_1(H)$. After the bias shift, $l(H)$ becomes equal to $l_2(H)$. Feature removal is typically done to improve learning efficiency because it reduces the size of the hypothesis space. However, identifying features to remove can be expensive and might even result in an incorrect bias, so the cost of removing features must be balanced against the savings.

Once we have chosen an $l(H)$, we can shift the procedural bias for that particular $l(H)$, assuming the chosen $l(H)$ stays fixed while we search the space of procedural biases for it. The search space $P(l_2(H))$, used for selecting $p(l_2(H))$, is shown in Figure 3 with two states, $p_1(l_2(H))$ and $p_2(l_2(H))$. A shift in procedural bias for the hypothesis space is a shift in the order of traversal of the hypotheses. Recall that the procedural bias has two components: $\phi(l(H))$, the operators (or algorithms) that can move the system from one state to another in the space, and $\theta(l(H))$, the partial ordering over the states. Together, these components enable the system to select a "next state." A shift in procedural bias normally alters the components individually. For example, if the set of operators for moving from one hypothesis to another includes generalization and specialization operators, then we could shift the procedural bias by removing one of the specialization operators. This might make some of the hypotheses (states) inaccessible. The evaluation function induces an order for state space traversal. For example, a simplicity preference combined with a hill climbing search process will concentrate on the portion of the state space dominated by simpler hypotheses. If we shift from a preference for simplicity to a preference for complexity, the focus will shift to a different portion of the search space. Note that although the procedural bias shift in Figure 3 results from a change in the set of hypothesis accessibility operators, an equivalent bias shift could have alternatively resulted from a change in the evaluation function (or from a change in the operators *and* the evaluation function).

5.3. Searching the meta-bias space

At the second tier (bias search space), we again face the question of whether we want the search to be static or dynamic. In most current systems, decisions at this level are static. However, in some systems there is a third level of flexibility. In this case, search also takes place at a meta-bias space level, which includes the search spaces $L(L(H))$, $L(P(l(H)))$, $P(l(L(H)))$, and $P(l(P(l(H))))$. At the meta-bias level, it is possible to select a procedural or representational bias for any search space in the second tier. Each state in a representational meta-bias space defines the states for a bias search space. In other words, it is an option for $l(L(H))$ or for $l(P(l(H)))$. Each state in a procedural meta-bias space is a different accessibility-ordering pair for the states in a bias space. In other words, it is an option for $p(l(L(H)))$ or for $p(l(P(l(H))))$.

To illustrate representational meta-bias space, suppose we wish to select a representational bias space (rather than a procedural bias space) dynamically. Thus we are selecting representational candidates $l_i(L(H))$ for $l(L(H))$. Then the meta-bias space is a search space in which a state is a set of candidate representational biases (e.g., $l_1(L(H))$ or $l_2(L(H))$ in Figure 4). This set provides choices for searching at the next tier down (i.e., for representational bias space search). If we want to increase the efficiency of the search through representational bias space, for example, we might move from one state in meta-bias space to another state that contains fewer alternative biases. An example is shown in Figure 4. In this figure, we start out with four representational biases. Each bias consists of a set of features for describing hypotheses. By moving to a state with only two biases, the efficiency of the search through bias space is improved. Once we have moved to state $l_2(L(H))$ in meta-bias space, we have chosen this state as the current state for $l(L(H))$. This results in a language for the bias search space $L(H)$ that has the same states that we see in $L(H)$ in Figure 2.

Next, we illustrate procedural meta-bias space. Suppose each state in this meta-bias space is an accessibility-ordering pair for searching a representational bias space. (Alternatively, it could be an accessibility-ordering pair for searching a procedural bias space.) Then each state in this meta-bias space imposes a particular accessibility mapping and partial ordering over the states for representational search at the next lower tier. Figure 5 shows an example. Here, we assume that $l_1(L(H))$ has been chosen for the current $l(L(H))$ and we are now searching for a procedural bias for $l_1(L(H))$, i.e., for $p(l_1(L(H)))$. If an order (state) is selected at this level, search at the next tier down (i.e., in representational bias space) will follow this order when searching through the biases $l_i(H)$. Again, the procedural bias selects both the operators and the partial ordering for the next lower level. Note that each state, $p_i(l_1(L(H)))$, of Figure 5 (if a partial ordering were included) is a procedural candidate for the space $L(H)$.

Conceptually, we could also imagine a fourth tier in addition to the three, and so on. Although we are not aware of any systems with more than three tiers, one can easily see how to extend the current framework to add tiers. Each additional tier adds the ability to explore alternative biases at the level below. This additional search can improve learning performance through increased efficiency or better predictions, but also adds computational costs that must be offset by the performance gain in order to have an overall increase in performance.

6. Recent research

Our framework for bias selection as search raises a number of important questions about research in this field:

- What does each search space (hypothesis space, bias space, meta-bias space) look like?
- Which tiers and dimensions are dynamic, and which are static?
- What evaluation method has been used to facilitate bias selection?

In addition, there are some general questions that can be asked when analyzing a learning system and its bias:

- What are the user's performance goals for the learner (e.g., accuracy, efficiency, readability)? How do they affect the decisions regarding the choice of tiers, evaluation method, static versus dynamic?
- How successful is the learner at meeting the user's performance goals?

The papers in this issue range from formal, theoretical analyses of bias-selection issues to empirical tests of particular methods. Most of the papers describe systems or analytical methods that examine multiple states in the second tier: that is, they search the bias space. Although none of these systems perform a completely automated search of the meta-bias space, several describe methods for searching the third tier by allowing the human designer to implement various search strategies in the second tier. Provost and Buchanan's paper is a good example of this.

The research in this area, presented in this issue and elsewhere, can be roughly characterized as either offline or online bias evaluation and selection. The former category consists of evaluation methods used by system designers to explore and select biases or bias search methods during the development stage of a learning system. The latter includes constructive induction, the use of prior knowledge to select biases, and other methods for searching the bias space dynamically during learning. Most of the papers in this issue describe offline learning methods: comparative studies (Ade et al., Stahl) or methods for encoding bias search strategies (Provost and Buchanan, Brodley). Turney introduces stability as a useful performance goal that could be used in conjunction with online or offline bias selection methods. Subramanian analyzes the use of the irrelevance principle as an online learning method. We discuss research on bias evaluation and selection, divided into offline and online techniques, in the following sections. The summary presented here is intended to be representative, not comprehensive.

6.1. Offline learning

Offline learning methods are evaluation methods used by researchers and designers to explore the effect of various biases and bias search methods on learning performance. Formal and empirical methods can be used to compare bias methods in various systems, giving us insight into the advantages and disadvantages of the different methods used. Ade et al. (this issue) present a comparative study of biases in three ILP (Inductive Logic Programming) systems – CLINT, GOLEM, and ITOU. Their NINA system allows a designer to specify bias for ILP learning declaratively, and it can shift biases in a prespecified sequence. NINA supports specification of the representational bias $l(H)$, which they call *syntactic bias,* and a form of procedural bias $p(l(H))$ that consists of a Boolean evaluation function for biases (this is termed *semantic bias.*) The performance goals are accuracy and efficiency; these are measured in an empirical comparison of the biases within the three systems.

Stahl (this issue) also compares different ILP approaches, presenting a formal analysis of when predicate invention is a useful shift. Predicate invention, a form of constructive induction, is a search mechanism through the representational bias space $L(H)$. The results of Stahl's analysis could be used at the procedural meta-bias level $P(l(L(H)))$ to determine whether predicate invention is a useful method to use for search at the bias level. In this case, the performance goal for defining "useful" is whether or not learning succeeds with the selected representational bias.

Other offline methods provide testbeds or paradigms for developing strategies for searching the bias space; these search methods can be used subsequently in online learning. Provost and Buchanan (this issue) describe a testbed for implementing *inductive policy*, that is, a declarative representation for a search method over the hypothesis space H and bias spaces $L(H)$ and $P(l(H))$. Their SBS (Search of the Bias Space) testbed, which has been implemented and is available for other researchers to use, allows a designer to explore the effects of various performance goals on the bias selection process. Therefore, a designer can use SBS to work within meta-bias space to make decisions about representational and procedural bias spaces. Thus their system is one in which the third (uppermost) tier of Figure 1 can be explored.

Brodley's Model Class Selection (this issue) is a learning system that demonstrates a particular paradigm for developing bias search strategies – heuristic rules are used to select and shift bias at a coarse grain size. The current implementation of MCS incorporates a set of heuristics to search the representational bias space $L(H)$. The procedural bias $p(l(H))$ for each representational bias $l(H)$ is thus selected and fixed at the outset of learning. However, one could envision using similar methods to search $P(l(H))$ as well as the (meta-bias) space of heuristic rules itself. Accuracy is the only performance goal that is considered in MCS.

Turney (this issue) describes the use of *stability* as a performance goal to be used in the evaluation of alternative representational and procedural biases. One could envision this measure being used at both the meta-bias and the bias space levels. A formal analysis of the stability measure is given. This measure can be used in online or offline learning. Offline analysis provides insight into how the use of this measure affects overall system performance; an online bias selection method could then incorporate this measure into the bias evaluation function to guide search in the representational and procedural bias spaces.

6.2. Online learning

For online learning, many alternative methods for searching and pruning the space of biases have been explored by machine learning researchers; research in this area is ongoing and active. For example, Subramanian's *irrelevance principle* (this issue) provides a formal criterion for determining when a shift of vocabulary bias is justified in problem solvers. In other words, the irrelevance principle guides the search through the representational bias space. We cannot immediately fit this work into the inductive learning framework of Figure 1 because the bias shift is performed in the context of speedup, rather than inductive, learning. Nevertheless, there is an obvious similarity between the space being searched by Subrananian's system and $L(H)$. The performance goals in Subramanian's work are accuracy and computational efficiency. Both a formal analysis of the irrelevance principle and empirical results of its application to learning domains are given.

The GABIL system of Spears and Gordon [34] uses cross-validation, a generate-and-test method, to select the degree to which the hypotheses are consistent with previously seen training data. The degree of hypothesis consistency is a choice of $p(l(H))$ because it is an evaluation over the hypotheses in the set $l(H)$. The performance goal is improved predictive accuracy. Spears and Gordon find that lower consistency is preferable when the data is noisy.

Gordon's PREDICTOR system [15] searches $L(H)$ by strengthening inductive bias whenever possible, and weakening bias minimally when necessary to restore bias correctness. The performance goal is to minimize the number of instances required to learn the target concept. PREDICTOR evaluates its bias using membership queries (i.e., requested instances [3]). This work reveals an important tradeoff between the reduced cost of learning with a stronger bias and the increased cost of more queries used to help in choosing a stronger bias. PREDICTOR performs feature selection. There has recently been an increasing interest in this topic, e.g., by Almuallim and Dietterich [2], Kira and Rendell [20], Vafaie and De Jong [37], Aha and Bankert [1], and John et al. [18].

The system of Bloedorn et al. [5] searches all three tiers of Figure 1, including meta-bias space. The performance goals are predictive accuracy, simplicity, and efficiency. Heuristics are used extensively in the bias selection process. The heuristics form a procedural meta-bias for decisions about the choice of representational bias and are thus decisions about $p(l(L(H))$. Evaluation of the meta-bias is a combination of generate-and-test, using meta-level examples, and suggestions from the user.

6.2.1. Constructive induction

Constructive induction methods are used to generate new terms in the hypothesis language. Bias shift occurs as a result of constructive induction both in the representational bias space $L(H)$ (if the new terms increase the expressiveness of the language, as with numerical ranges or disjunctions in a language that does not otherwise permit disjunctions) and in the procedural bias space $P(H)$ (for example, the new terms may make certain theories simpler, so if simple theories are preferred, then the preference order is changed).

Constructive induction is an active field of research and the most widely studied form of bias change, as evidenced by the recent workshops and journal issues devoted to this topic. Fawcett chaired a workshop at the 1994 Machine Learning Conference entitled “Constructive Induction and Change of Representation” at which several novel approaches to constructive induction were presented. This followed a 1991 Machine Learning workshop on “Constructive Induction” chaired by Matheus.

Several papers in a recent special issue of *Machine Learning* on “Evaluating and Changing Representation” examine constructive induction. For example, in that issue Wnek and Michalski [39], Wrobel [41], and Kietz and Morik [19] describe methods for dynamically shifting bias by performing constructive induction when learning fails. Furthermore, those papers address some of the same issues we focus on here.

Rendell has done much work on this topic. The performance goals of Rendell’s [27] constructive induction system are speed, accuracy, and conciseness (simplicity). The constructive induction method used is “peak merging.” New peaks (hypothesis disjuncts) are formed by constructing new terms (attributes). Each new peak is comprised of multiple old peaks.

As mentioned at the beginning of this section, a pertinent question to ask is, “What evaluation method facilitates bias selection?” Matheus [21] presents a framework in which he elaborates sources upon which to base the evaluation of a bias shift involving feature construction (e.g., peak merging). These sources are the instances, the hypotheses, and the domain knowledge.

6.2.2. Prior knowledge

Prior knowledge can be used to reduce the search space and guide the selection of representational and procedural biases. Learning and modifying this prior knowledge thus amounts to searching the meta-bias level. Although none of the papers in this issue focus on this topic, it is an important and active area of research. In 1992, desJardins chaired a AAAI workshop entitled "Constraining Learning with Prior Knowledge" that explored novel approaches for using existing knowledge to reduce the computational complexity of the learning problem, including the evaluation and selection of biases.

Russell and Grosof [29] first introduced the concept of *declarative bias*, an explicit specification of the representational bias $l(H)$. The advantage of declarative bias is that because the bias is represented explicitly, it is easy to analyze and modify it when necessary. Cohen's GRENDEL [12] allows a user to specify bias declaratively in a FOIL-like inductive learning system. The declarative bias allows the system to make use of a variety of types of prior knowledge directly and explicitly.

desJardins [14] describes a method for using background knowledge to evaluate and select biases dynamically with respect to expected accuracy and learning time in a decision-theoretic framework. This method provides an evaluation function over the space of representational biases $L(H)$, and could be extended to evaluate procedural biases as well. Baltes and MacDonald [4] and Datta and Kibler [13] also describe methods that use previously learned concepts (prior knowledge) to bias the learning of new concepts.

The use of prior knowledge to bias genetic algorithms and neural networks have recently been studied by several researchers. The systems of Gordon and Subramanian [16], as well as Schultz and Grefenstette [32], use prior knowledge to initialize a genetic algorithm. The former system uses high-level advice for the initialization. KBANN [35] uses a domain theory in the form of propositional rules to initialize a neural network. Pratt [24] describes the *discriminability-based transfer* method for incorporating knowledge acquired during previous learning tasks into a neural network.

7. Conclusions

The framework we have presented for treating bias selection as search has two essential features: bias is divided into representational and procedural components, and learning systems are characterized as potentially having multiple tiers of search (in hypothesis space, bias space, meta-bias space, and so on). We advocate the use of this framework both as a tool for understanding, analyzing, and comparing existing systems, and as a basis for guiding the design and development of new machine learning systems. We believe that doing analysis and development within this framework will lead to a better understanding of where the implicit and explicit biases in learning systems lie, will reduce both system development and knowledge engineering time and effort, and will allow explicit representation and incorporation of considerations such as background knowledge and cost.

Of course, the framework provides only the skeleton of a design and an abstract formalism for representing and analyzing bias as a multiple-tiered search process. The real work will come in the development of systems that fill in the components of the outline we have sketched, and that are able to perform learning in complex and dynamic domains. The papers in this issue begin to address some of the critical questions in this development; we hope that this special issue of *Machine Learning* inspires the research community to look more closely at the central issue of the evaluation and selection of biases in machine learning systems.

Acknowledgments

We would like to thank Bill Spears and Pat Langley for insightful questions and interesting discussions on this topic, and Bill Spears, Rob Holte, Haym Hirsh, William Cohen, Paul Utgoff, David Aha, Lise Getoor, and Tom Dietterich for proofreading and commenting on earlier drafts. Their comments improved both the style and content of this paper. We also thank the reviewers for their thoughtful reviews of the papers in this issue.

Notes

1. Biases can also affect the definition or selection of instances (see Saxena [30]). We omit a discussion of this topic for the sake of brevity.
2. The arrows in Figure 1 go downward only. This is a simplification to clarify our presentation. Bias revision is typically both data- and model-driven, so that the lower tiers influence the higher tiers and vice versa.

References

1. Aha, D. and Bankert, R. (1994). Feature selection for case-based classification of cloud types. In *Proceedings of the 1994 Workshop on Case-Based Reasoning*, pages 106–112. AAAI Press.
2. Almuallim, H. and Dietterich, T. (1991). Learning with many irrelevant features. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 547–552. AAAI Press.
3. Angluin, D. (1988). Queries and concept learning. *Machine Learning*, 2(4):319–342.
4. Baltes, J. and MacDonald, B. (1992). Case-based meta learning: Sustained learning supported by a dynamically biased version space. In *Proceedings of the ML92 Workshop on Biases in Inductive Learning*.
5. Bloedorn, E., Michalski, R., and Wnek, J. (1993). Multistrategy constructive induction: AQ17-MCI. In *Proceedings of the Second International Workshop on Multistrategy Learning*, pages 188–206.
6. Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. (1987). Occam's razor. *Information Processing Letters*, 24:377–380.
7. Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. (1989). Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association for Computing Machinery*, 36(4):929–965.
8. Cardie, C. (1993). Using cognitive biases to guide feature set selection. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society*, pages 743–748. Lawrence Erlbaum Associates.
9. Chaitin, G. J. (1977). Algorithmic information theory. *IBM J. Res. Develop.*, 21:350–359.
10. Chrisman, L. (1989). Evaluating bias during PAC-learning. In *Machine Learning Workshop*, pages 469–471. Morgan Kaufmann.

11. Cobb, H. (1992). Inductive biases in a reinforcement learner. In *Proceedings of the ML92 Workshop on Biases in Inductive Learning.*
12. Cohen, W. W. (1995). Grammatically biased learning: Learning Horn theories using an explicit antecedent description language. *Artificial Intelligence.*
13. Datta, P. and Kibler, D. (1992). Utilizing prior concepts for learning. In *Proceedings of the ML92 Workshop on Biases in Inductive Learning.*
14. desJardins, M. (1994). Evaluation of learning biases using probabilistic domain knowledge. In *Computational Learning Theory and Natural Learning Systems, vol. 2*, chapter 7, pages 95–112. The MIT Press.
15. Gordon, D. (1990). *Active Bias Selection for Incremental, Supervised Concept Learning.* PhD thesis, University of Maryland, Department of Computer Science. Also available as Technical Report UMIACS-TR-90-60 CS-TR-2464.
16. Gordon, D. and Subramanian, D. (1993). A multistrategy learning scheme for agent knowledge acquisition. *Informatica*, 17:331–346.
17. Hirsh, H. (1990). Knowledge as bias. In *Change of Representation and Inductive Bias.* Kluwer Academic Publishers.
18. John, G., Kohavi, R., and Pfleger, K. (1994). Irrelevant features and the subset selection problem. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 121–129. Morgan Kaufmann.
19. Kietz, J. and Morik, K. (1994). A polynomial approach to the constructive induction of structured knowledge. *Machine Learning*, 14(2):193–218.
20. Kira, K. and Rendell, L. (1992). A practical approach to feature selection. In *Proceedings of the Ninth International Conference on Machine Learning*, pages 249–256. Morgan Kaufmann.
21. Matheus, C. (1991). The need for constructive induction. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 173–177. Tioga.
22. Michalski, R. (1983). A theory and methodology of inductive learning. In Michalski, R., Carbonell, J., and Mitchell, T., editors, *Machine Learning I*, pages 83–134. Tioga.
23. Mitchell, T. (1980). The need for biases in learning generalizations. Technical Report CBM-TR-117, Rutgers University.
24. Pratt, L. Y. (1993). Discriminability-based transfer between neural networks. In Giles, C. L., Hanson, S. J., and Cowan, J. D., editors, *Advances in Neural Information Processing Systems 5*, pages 204–211. Morgan Kaufmann Publishers, San Mateo, CA.
25. Provost, F. J. and Buchanan, B. (1992). Inductive policy. In *AAAI-92*, pages 255–261. AAAI Press/The MIT Press.
26. Rendell, L. (1987). Similarity-based learning and its extensions. *Computational Intelligence*, 3:241–266.
27. Rendell, L. (1990). Feature construction for concept learning. In *Change of Representation and Inductive Bias*, pages 327–353. Kluwer.
28. Rendell, L., Seshu, R., and Tcheng, D. (1987). More robust concept learning using dynamically-variable bias. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 66–78. Morgan Kaufmann.
29. Russell, S. and Grosof, B. (1987). A declarative approach to bias in concept learning. In *AAAI*, pages 505–510.
30. Saxena, S. (1991). On the effect of instance representation on generalization. In *Proceedings of the Eighth International Workshop on Machine Learning*, pages 198–202. Morgan Kaufmann.
31. Schaffer, C. (1994). A conservation law for generalization performance. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 259–265. Morgan Kaufmann.
32. Schultz, A. and Grefenstette, J. (1990). Improving tactical plans with genetic algorithms. In *Proceedings of the IEEE Conference on Tools for AI*, pages 328–334. IEEE Press.
33. Solomonoff, R. J. (1964). A formal theory of inductive inference. *Information and Control*, 7.
34. Spears, W. and Gordon, D. (1992). Is consistency harmful? In *Proceedings of the ML92 Workshop on Biases in Inductive Learning.*
35. Towell, G., Shavlik, J., and Noordewier, M. (1990). Refinement of approximate domain theories by knowledge-based neural networks. In *Proceedings of AAAI-90*, pages 861–866. Morgan Kaufmann.

36. Utgoff, P. (1986). Shift of bias for inductive concept learning. In Michalski, R., Carbonell, J., and Mitchell, T., editors, *Machine Learning II*, pages 107–148. Morgan Kaufman.
37. Vafaie, H. and Jong, K. D. (1993). Robust feature selection algorithms. In *Proceedings of the Fifth Conference on Tools for Artificial Intelligence*, pages 356–363. IEEE Computer Society Press.
38. Vapnik, V. and Chervonenkis, A. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280.
39. Wnek, J. and Michalski, R. (1994). Hypothesis-driven constructive induction in AQ17-HCI: A method and experiments. *Machine Learning*, 14(2):139–168.
40. Wolpert, D. (1992). On the connection between in-sample testing and generalization error. *Complex Systems*, 6:47–94.
41. Wrobel, S. (1994). Concept formation during interactive theory revision. *Machine Learning*, 14(2):169–192.